

**SeUng Jung**

**Data Life Cycle on AWS**

*Lab 4-1 Exploratory Data Analysis with*

*Spark and Zeppelin*

**Table of Contents**

Lab 설명 3

Lab Architecture 3

Exploratory Data Analysis with Spark and Zeppelin 4

# Lab 설명

이번 Lab은 AWS의 관리형 하둡 플랫폼인 EMR에서 Spark, Zeppelin 서비스를 통해 S3 데이터 레이크에 저장되어 있는 데이터를 분석합니다.

Glue를 통해 생성된 Data Catalog는 Hadoop의 Hive metastore와 호환이 가능하여 손쉽게 Spark에서 데이터를 로딩할 수 있습니다.

Zeppelin은 Spark와 함께 사용하기 유용한 웹 노트북 서비스로, EMR 클러스터 생성시 체크박스에 선택만으로 쉽게 설치하여 활용할 수 있습니다. 단순 명령어 실행에서 간단한 Visualization기능까지 포함하고 있어 이번 실습은 주로 Zeppelin을 활용 할 것입니다.

# Lab Architecture



# Exploratory Data Analysis with Spark and Zeppelin

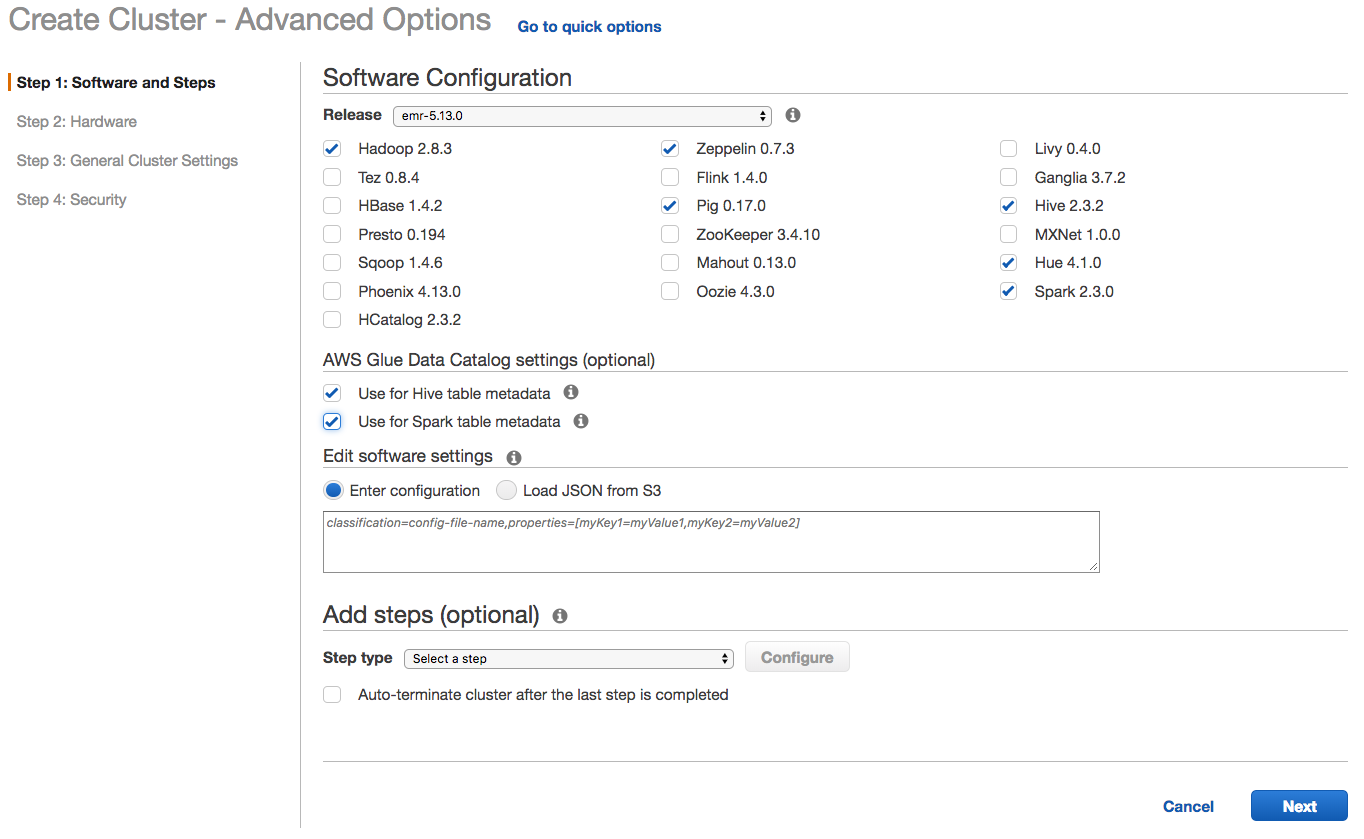
Exploratory Data Analysis는 데이터 분석의 가장 초기 단계에 데이터의 특성을 파악하기 위해서 간단한 쿼리와 그래프를 활용하는 분석 프로세스 입니다.

먼저 Spark, Zeppelin이 포함된 EMR 클러스터를 생성하고, Glue Data Catalog에 접근하여 데이터를 로딩하여, Zeppelin 웹 노트북에서 쿼리와 그래프를 그려봅니다.

1. 우측 상단의 리전을 **Oregon (US-WEST)**으로 선택합니다.
2. AWS Management Console에서 EMR서비스 화면으로 이동, **[Create cluster]**를 선택합니다.
3. 상단에 **[Go to advanced options]**를 선택합니다.

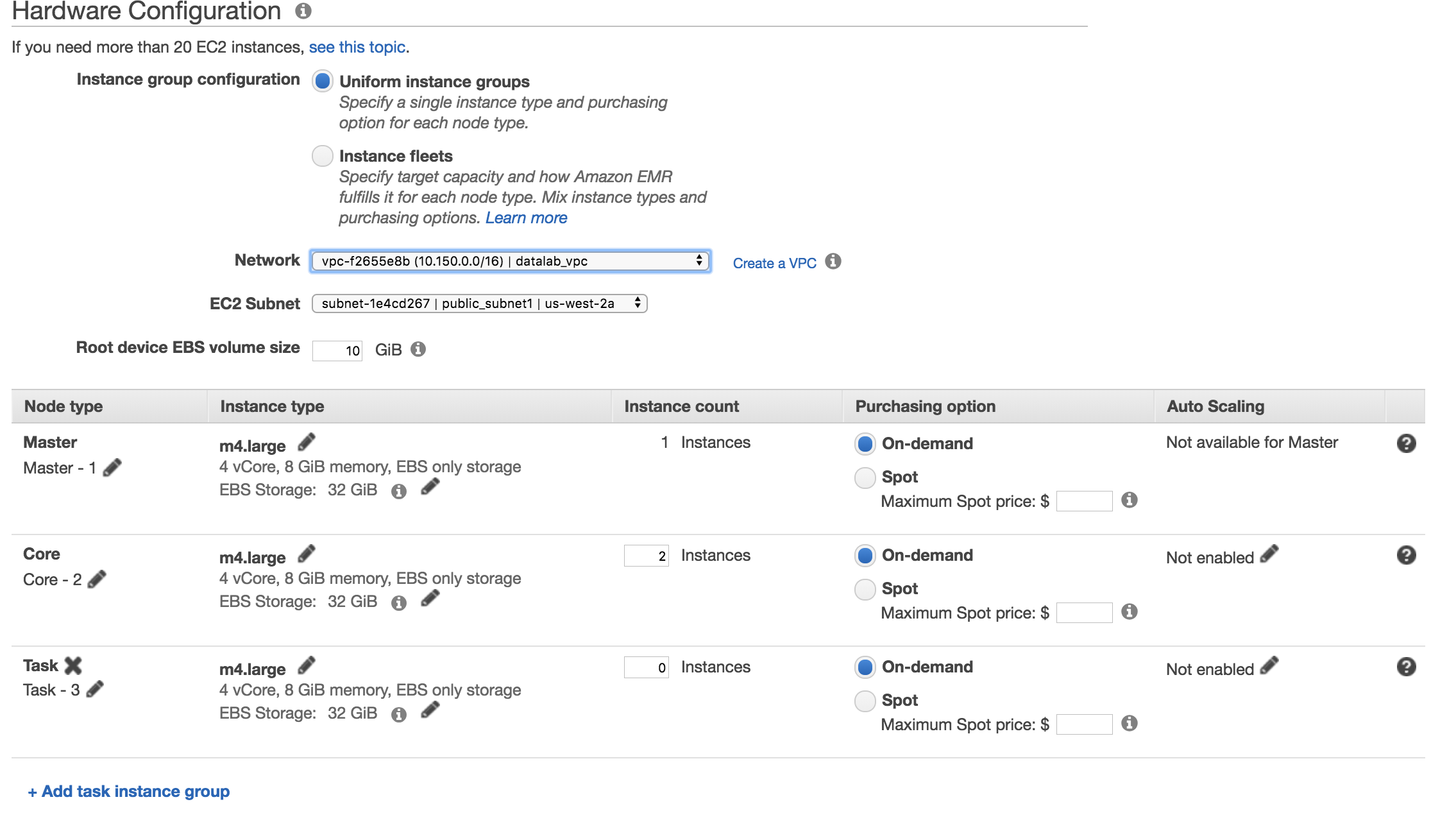


1. 소프트웨어에서는 **Zeppelin**, **Spark**를 선택해주고, **AWS Glue Data Catalog settings**에서 모두 체크 표시 합니다.

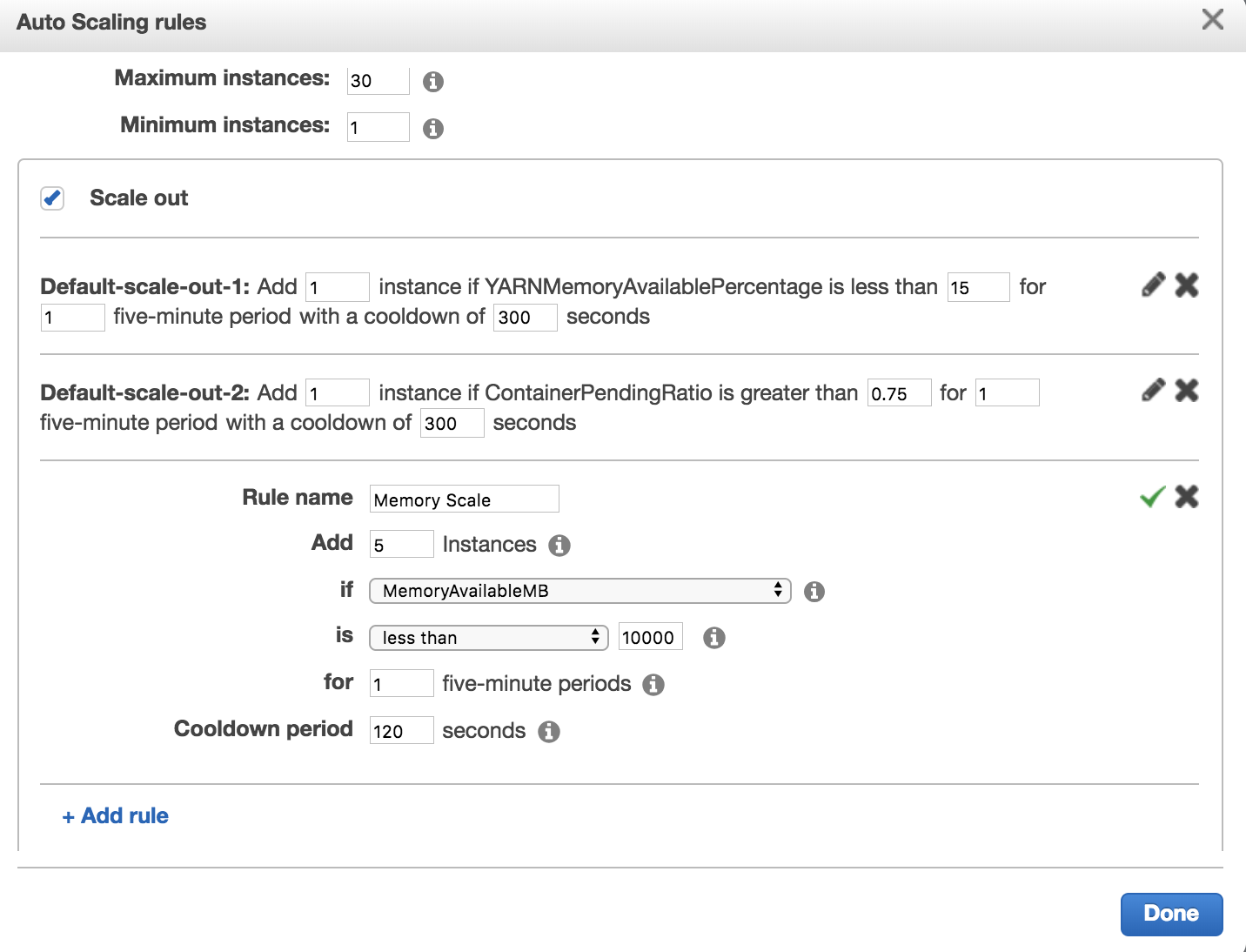


1. Hardware Configuration에서 Network의 VPC를 **[datalab\_vpc]**를 선택합니다.

그리고 Task 노드의 **auto-scaling** 을 눌러 설정합니다.

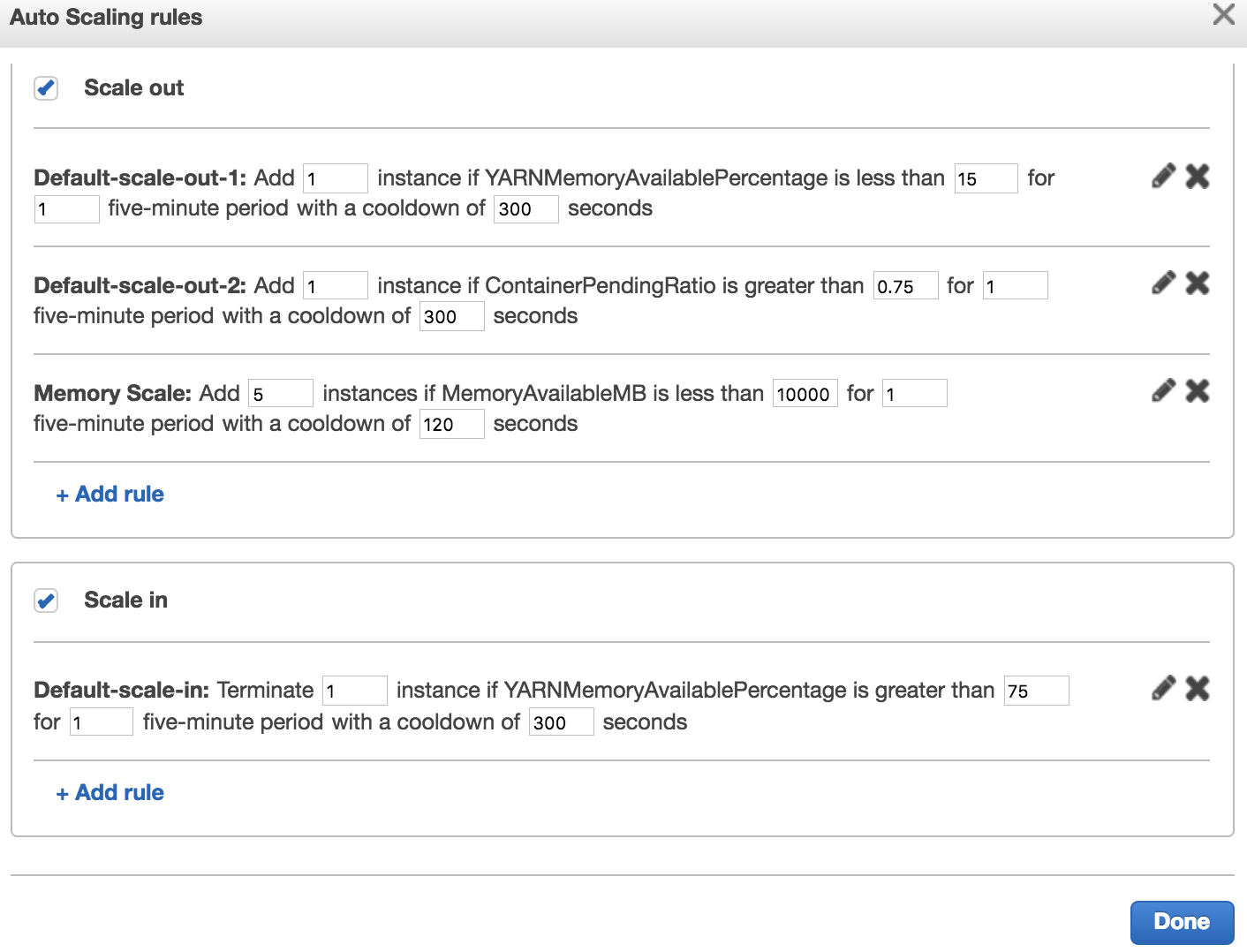


1. 대규모의 작업이 진행되면 Task node가 Auto scaling 되도록 설정할 수 있습니다. 최대 30대까지 확장되도록 설정하고 Scale out / in 설정을 확인합니다.

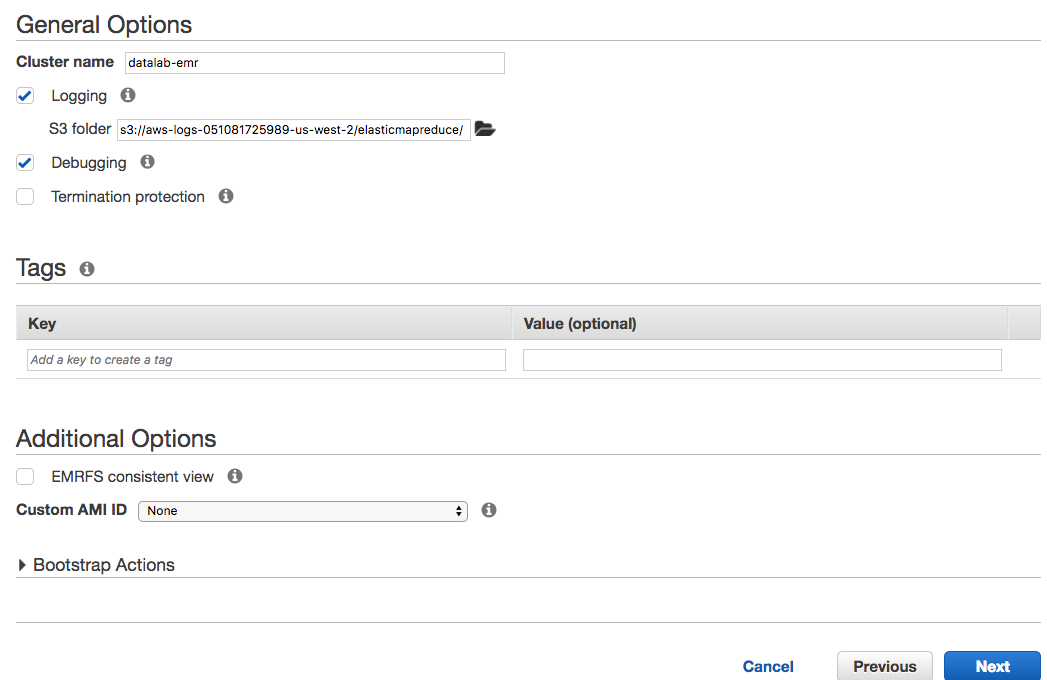


1. 기본적인 룰 이외에 Spark는 메모리에 민감하게 동작하므로 메모리 부족시 클러스터 확장 룰을 추가합니다.

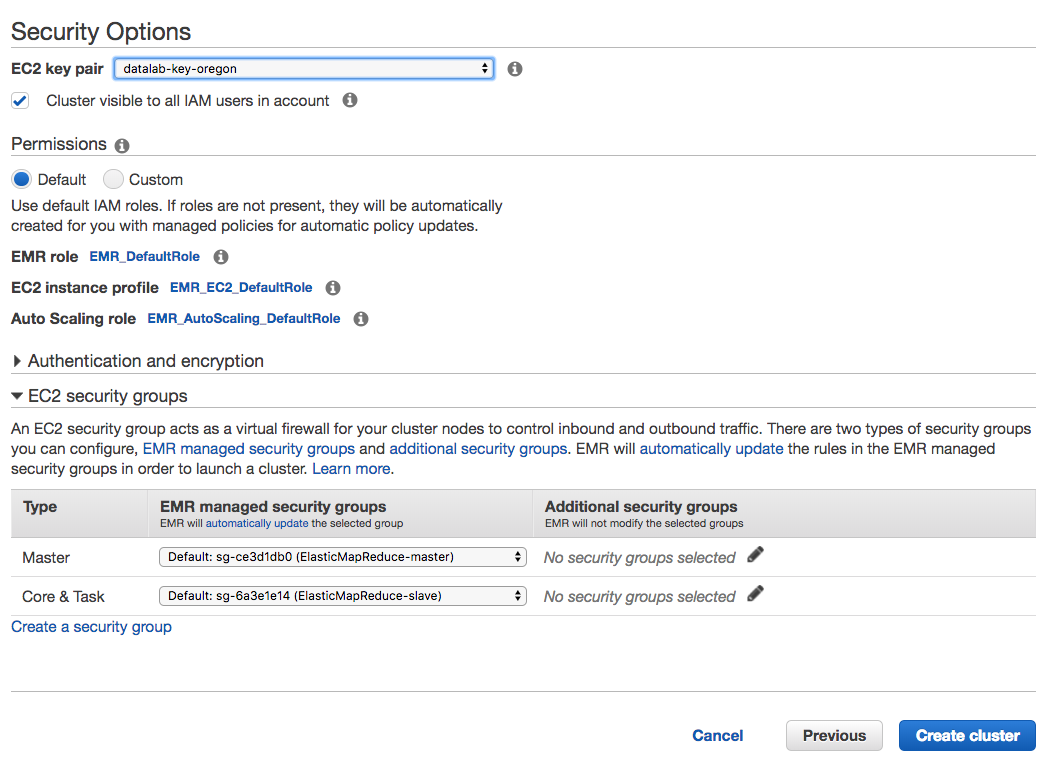
룰을 추가한 이후에 초록색 v 체크버튼을 눌러야 룰에 추가됩니다.



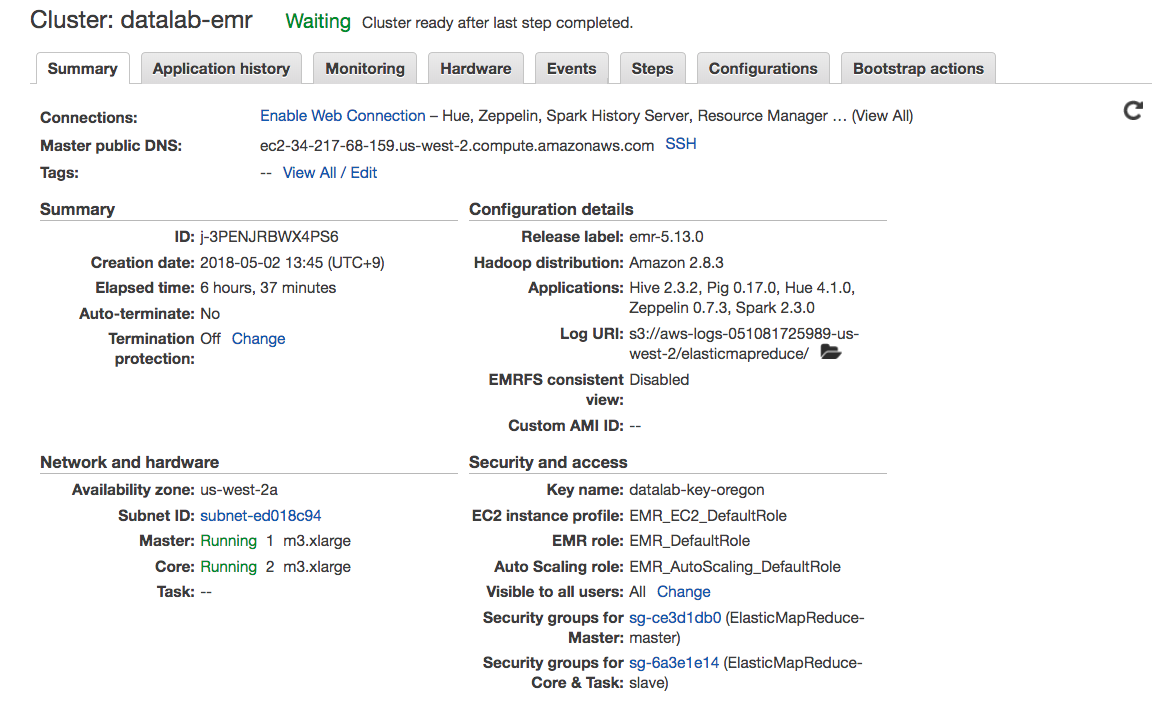
1. General Options에서는 클러스터 이름 **datalab-emr**을 입력하고, **Termination protection**은 체크 해제 합니다.



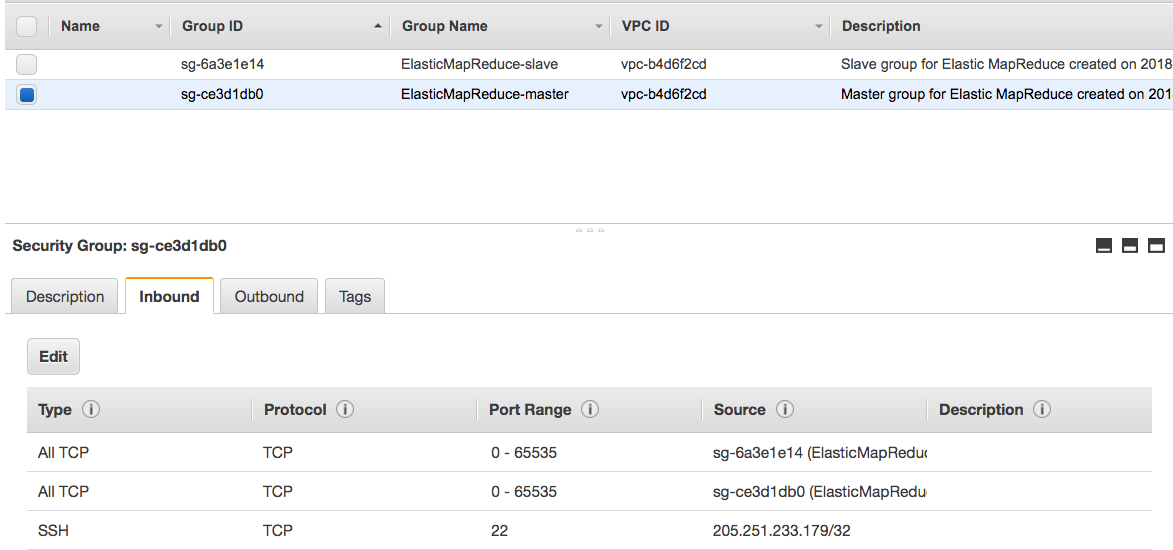
1. 키 페어를 선택하고 클러스터를 생성합니다. 이후 Zeppelin 접속을 위해 반드시 실습 처음에 생성한 키페어 선택을 해줘야 합니다.



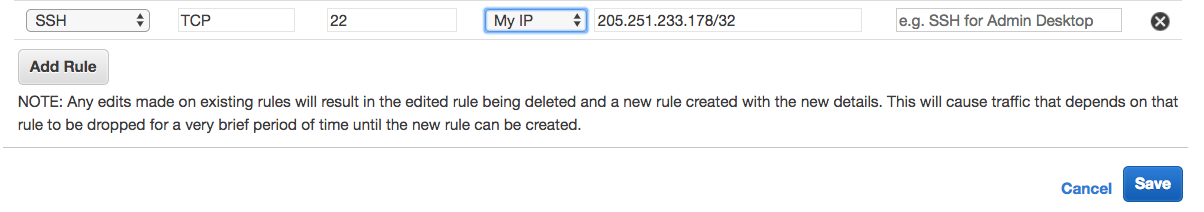
1. 다음처럼 클러스터 생성이 완료되면 **Waiting** 상태로 변경됩니다. EMR 클러스터 생성에는 약 15분 정도 소요됩니다.



1. **Master public DNS**의 URL 을 별도의 메모장에 복사해 놓습니다.
2. 아래쪽 Security group for Master의 링크를 통해 설정화면으로 들어갑니다.
3. Security Group Name이 **ElasticMapReduce-master**인지 확인하고 아래쪽 **[Inbound]** Tab을 선택합니다.



1. **[Edit]** 를 누르고 **[Add Rule]** 하여 **SSH** 에 대한 **My IP** 접근을 추가합니다.



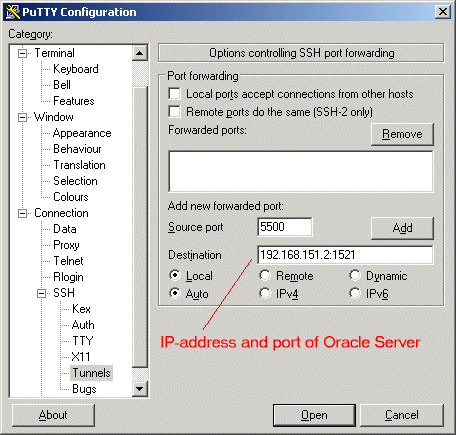
1. 터미널을 실행시켜 key가 저장된 폴더에서 다음 명령어를 실행합니다. 키이름과 DNS이름은 각자 환경에 맞는 이름으로 변경하셔야 합니다.

ssh -i [KeyPair] -N -L 8157:localhost:8890 hadoop@[**Master Node DNS**]

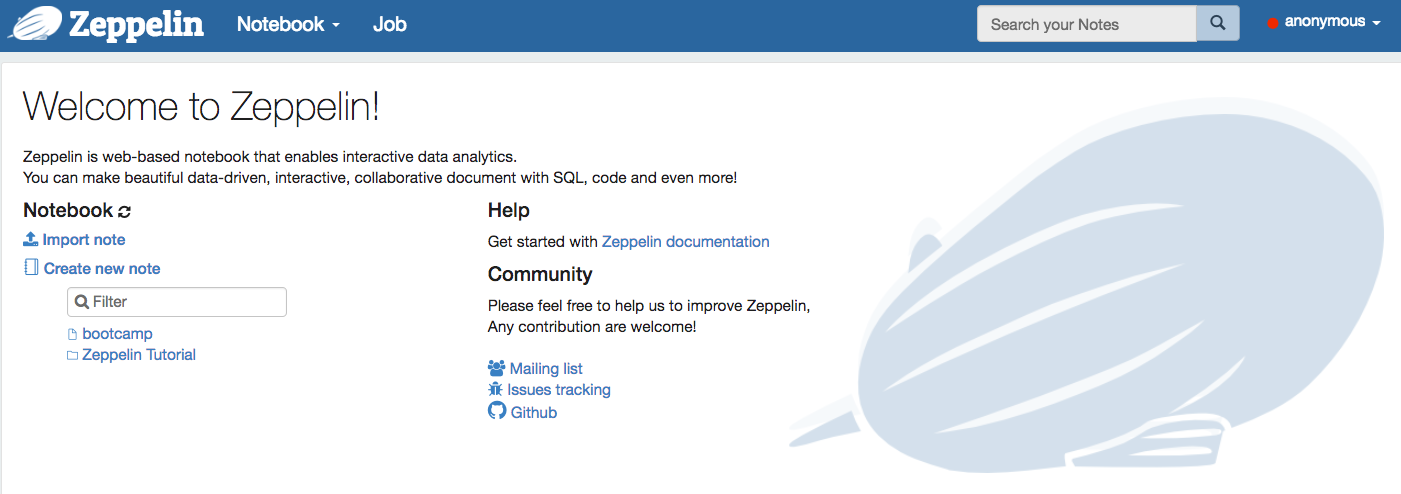
|  |
| --- |
| ssh -i datalab-key-oregon.pem -N -L 8157:localhost:8890 [hadoop@ec2-34-217-68-159.us-west-2.compute.amazonaws.com](mailto:hadoop@ec2-34-217-68-159.us-west-2.compute.amazonaws.com) |

Note : 윈도우 유저는 PuTTY에 아래 항목에 값을 입력합니다.

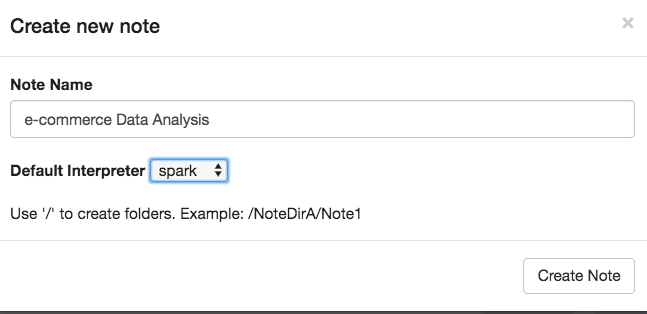
Source : 8157 / Destination : localhost:8890 / username : hadoop



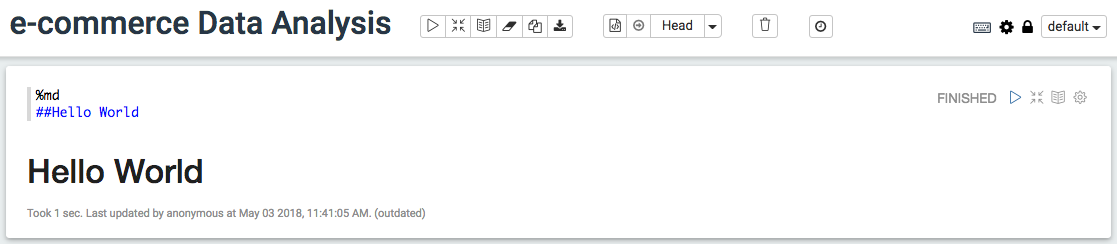
1. 접속이 완료되면 브라우저를 실행합니다.
2. 브라우저에 <http://localhost:8157/> 를 입력하면 다음 화면을 볼 수 있습니다.



1. 화면 중간에 **[Create new note]**를 누르고 Note Name에 **[e-commerce Data Analysis]** 이름을 입력하면 새로운 노트북이 생성됩니다.



1. 다음과 같이 노트북이 생성됩니다. 만약 Zeppelin 사용에 익숙한 유저는 바로 25번 실습으로 넘어가도 좋습니다.
2. 코드를 입력하고 화살표 버튼을 누르거나 **shift+enter**를 하면 실행이 됩니다.

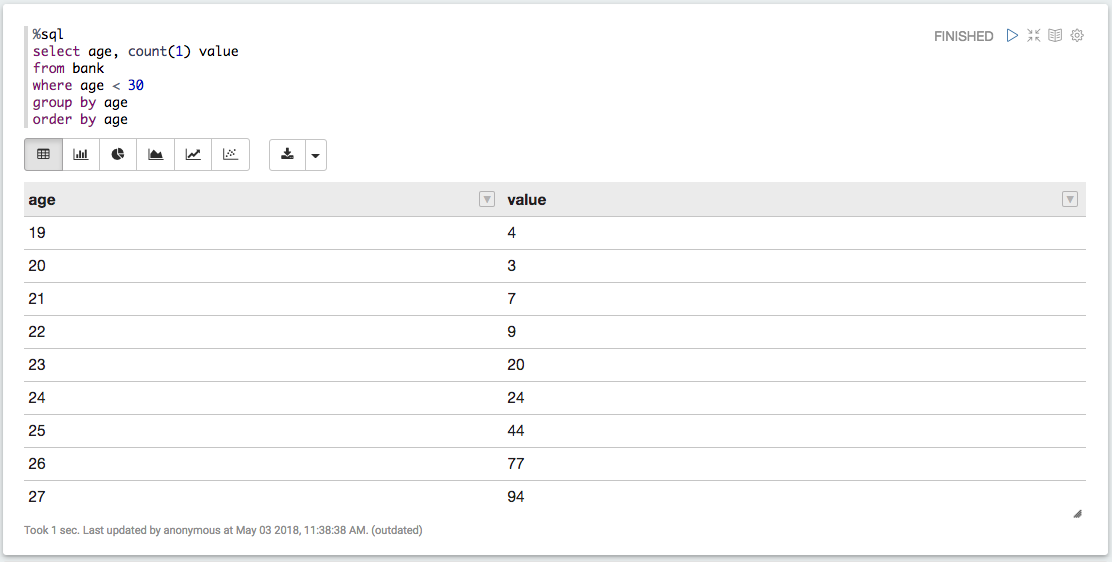


1. 간단한 테스트를 위해 아래의 튜토리얼 코드를 입력, 실행합니다.

|  |
| --- |
| import org.apache.commons.io.IOUtils  import java.net.URL  import java.nio.charset.Charset  // Zeppelin creates and injects sc (SparkContext) and sqlContext (HiveContext or SqlContext)  // So you don't need create them manually  // load bank data  val bankText = sc.parallelize(  IOUtils.toString(  new URL("https://s3.amazonaws.com/apache-zeppelin/tutorial/bank/bank.csv"),  Charset.forName("utf8")).split("\n"))  case class Bank(age: Integer, job: String, marital: String, education: String, balance: Integer)  val bank = bankText.map(s => s.split(";")).filter(s => s(0) != "\"age\"").map(  s => Bank(s(0).toInt,  s(1).replaceAll("\"", ""),  s(2).replaceAll("\"", ""),  s(3).replaceAll("\"", ""),  s(5).replaceAll("\"", "").toInt  )  ).toDF()  bank.registerTempTable("bank") |

1. 다음 쿼리를 입력해서 데이터를 조회합니다. 결과값이 보이면 쿼리 아래의 막대 그래프를 선택해보면 데이터에 맞는 그래프가 보여집니다.

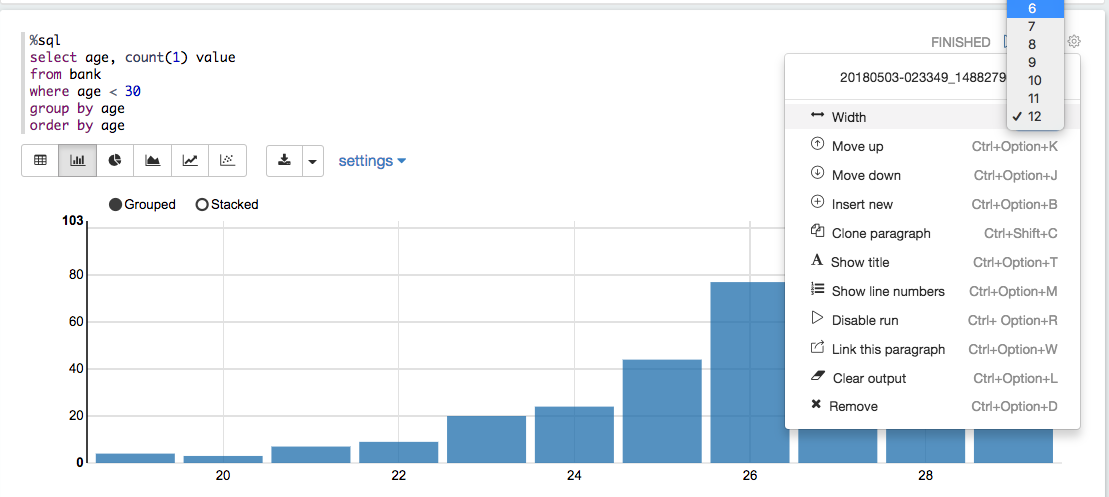
|  |
| --- |
| **%**sql  select age, count(1) value  from bank  where age < 30  group by age  order by age |

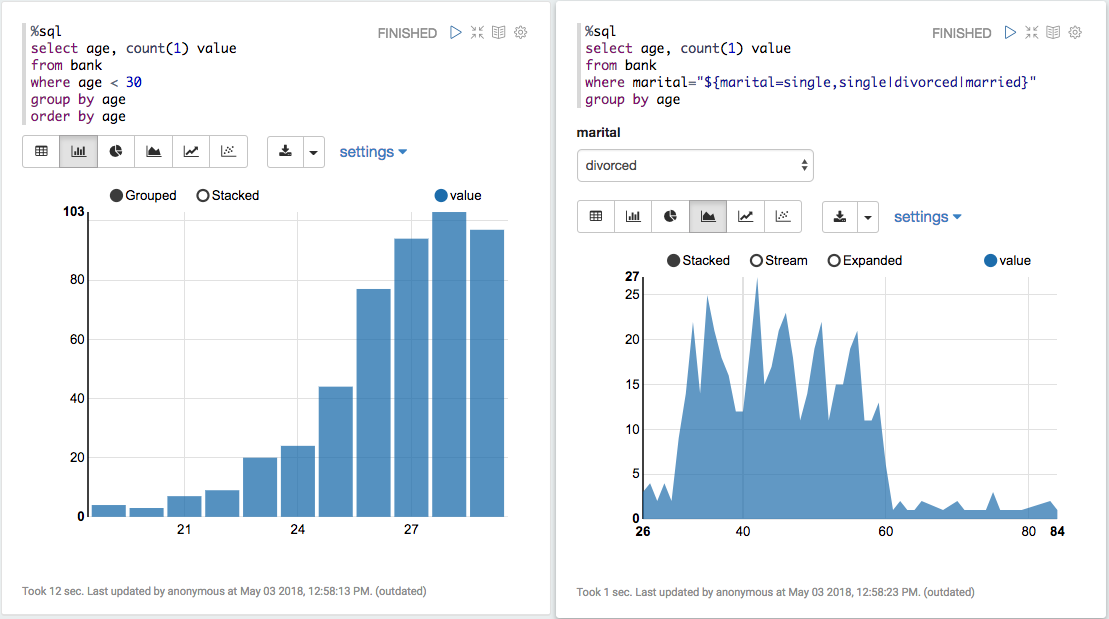


1. 추가로 한개의 쿼리를 더 테스트 해봅니다.

|  |
| --- |
| %sql  select age, count(1) value  from bank  where marital="${marital=single,single|divorced|married}"  group by age |

1. 각각의 명령 실행 창의 크기를 조절하여 함께 비교해볼 수 있습니다.

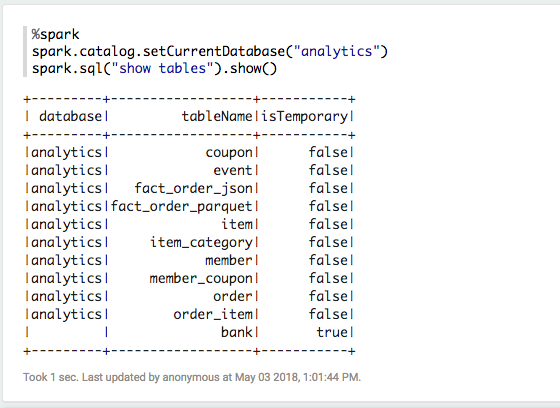




1. 지금부터 Glue Data Catalog를 사용하여 S3 Analytics 버킷에 저장되어 있는 데이터를 분석해보겠습니다.

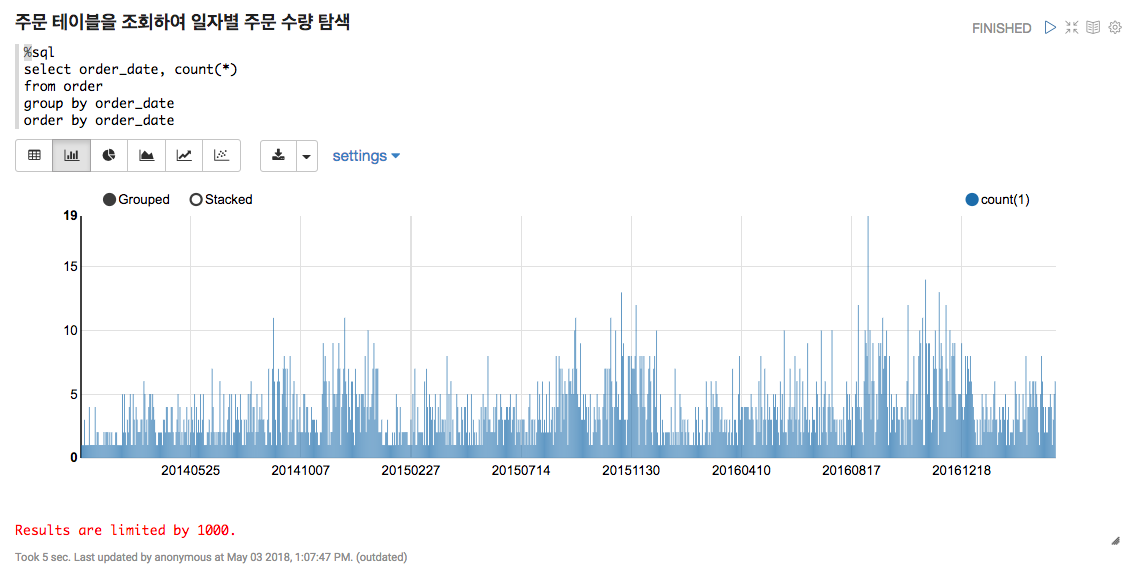
|  |
| --- |
| %spark  spark.sql("show databases").show() |

|  |
| --- |
| %spark  spark.catalog.setCurrentDatabase("analytics")  spark.sql("show tables").show() |



1. Glue Data Catalog에 저장된 것과 동일하게 테이블 형태로 데이터 접근이 가능합니다. 다음 쿼리를 실행하고 막대 그래프로 변경해서 시각화 해보시기 바랍니다.

|  |
| --- |
| %sql  select order\_date, count(\*)  from order  group by order\_date  order by order\_date |

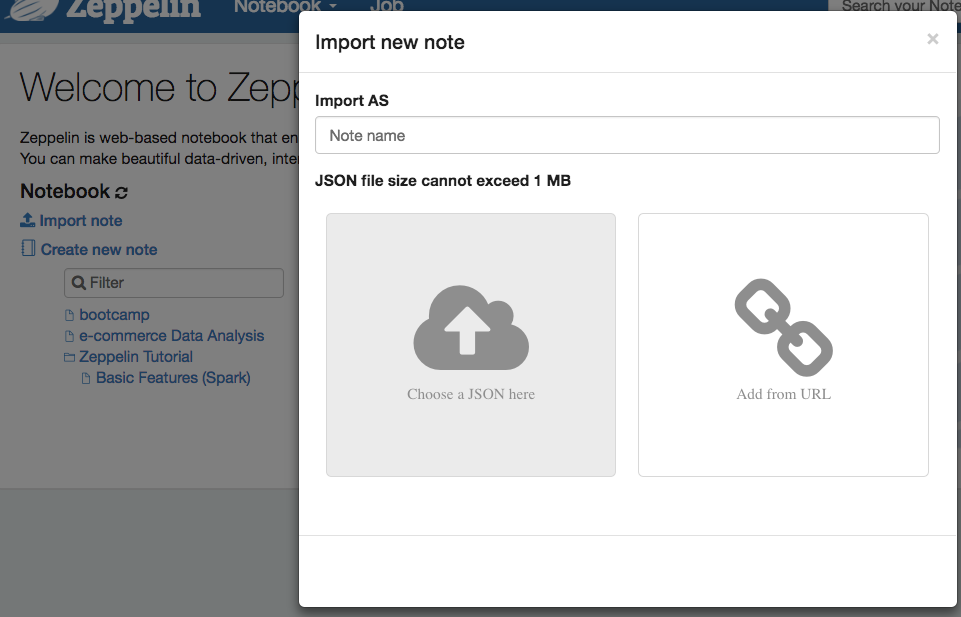


1. Zeppelin 노트북 활용이 익숙해졌다면 샘플 노트북을 다운 받아 import 시킵니다.

<https://s3-us-west-2.amazonaws.com/datalab-docs/etl_code/e-commerce+Data+Analysis.json>

(JSON 파일이므로 직접 다운로드 안되면, 내용 복사하여 편집기에서 .json으로 다시 저장)

1. Zeppelin 초기 화면에서 **[Import note]** 선택, **[Choose a JSON here]** 선택하여 노트 생성



1. 여기까지 실행하여 새로운 노트가 생성되면 앞에서 수행한 명령어 이후로 단계별로 실행해 봅니다.